Agree to Disagree: Robust Anomaly Detection with Noisy Labels

DENNIS M. HOFMANN, Worcester Polytechnic Institute, USA
PETER M. VANNOSTRAND, Worcester Polytechnic Institute, USA
LEI MA, Worcester Polytechnic Institute, USA
HUAYI ZHANG*, ByteDance, USA and Worcester Polytechnic Institute, USA
JOSHUA C. DEOLIVEIRA, Worcester Polytechnic Institute, USA
LEI CAO, University of Arizona, USA and Massachusetts Institute of Technology, USA
ELKE A. RUNDENSTEINER, Worcester Polytechnic Institute, USA

Due to the scarcity of reliable anomaly labels, recent anomaly detection methods leveraging noisy auto-generated labels either select clean samples or refurbish noisy labels. However, both approaches struggle due to the unique properties of anomalies. *Sample selection* often fails to separate sufficiently many clean anomaly samples from noisy ones, while *label refurbishment* erroneously refurbishes *marginal* clean samples. To overcome these limitations, we design Unity, the *first* learning from noisy labels (LNL) approach for anomaly detection that elegantly leverages the merits of both sample selection and label refurbishment to iteratively prepare a diverse clean sample set for network training. Unity uses a pair of deep anomaly networks to collaboratively select samples with clean labels based on prediction agreement, followed by a disagreement resolution mechanism to capture marginal samples with clean labels. Thereafter, Unity utilizes unique properties of anomalies to design an anomaly-centric contrastive learning strategy that accurately refurbishes the remaining noisy labels. The resulting set, composed of *selected and refurbished* clean samples, will be used to train the anomaly networks in the next training round. Our experimental study on 10 real-world benchmark datasets demonstrates that Unity consistently outperforms state-of-the-art LNL techniques by up to 0.31 in F-1 Score (0.52 → 0.83).

CCS Concepts: • **Computing methodologies** → **Anomaly detection**; • **Information systems** → **Data cleaning**.

Additional Key Words and Phrases: Anomaly Detection, Noisy Labels, Robust Machine Learning.

ACM Reference Format:
Dennis M. Hofmann, Peter M. VanNostrand, Lei Ma, Huayi Zhang, Joshua C. DeOliveira, Lei Cao, and Elke A. Rundensteiner. 2025. Agree to Disagree: Robust Anomaly Detection with Noisy Labels. *Proc. ACM Manag. Data* 3, 1 (SIGMOD), Article 7 (February 2025), 24 pages. https://doi.org/10.1145/3709657

*Research conducted while a PhD Candidate at Worcester Polytechnic Institute

Authors' Contact Information: Dennis M. Hofmann, dmhofmann@wpi.edu, Worcester Polytechnic Institute, USA; Peter M. VanNostrand, pvannostrand@wpi.edu, Worcester Polytechnic Institute, USA; Lei Ma, lma5@wpi.edu, Worcester Polytechnic Institute, USA; Huayi Zhang, hzhang4@wpi.edu, ByteDance, USA and Worcester Polytechnic Institute, USA; Joshua C. DeOliveira, jcdeoliveira@wpi.edu, Worcester Polytechnic Institute, USA; Lei Cao, University of Arizona, USA, caolei@arizona.edu and Massachusetts Institute of Technology, USA; Elke A. Rundensteiner, rundenst@wpi.edu, Worcester Polytechnic Institute, USA.

ACM 2836-6573/2025/2-ART7
https://doi.org/10.1145/3709657

1 Introduction

Anomaly detection, aiming to identify samples that significantly differ from the expected behavior, is critical for tasks such as detecting financial fraud, system malfunctions, cybersecurity attacks, and life-threatening health conditions [8, 14, 15, 21, 29].

Due to the rarity of anomalies and thus lack of high-quality anomaly labels, anomaly detection traditionally used to be unsupervised [14, 43]. Unfortunately, unsupervised techniques tend to perform significantly worse than their supervised counterparts [4, 14]. Recent works thus use unsupervised anomaly detectors for generating initial pseudo-labels that can then aid in training supervised deep-learning models to classify anomalies [3, 17, 40, 45]. However, generated pseudo-labels tend to be *noisy* (inaccurate), which degrades network performance [38, 42]. This leads recent methods to propose Learning from Noisy Labels (LNL) [38].

Challenges Illustrated With Real-World Datasets. Next, we illustrate the challenges faced when learning from noisy labels for anomaly detection (LNL-AD) in contrast to classification (LNL-C) using the Annthyroid and Mnist benchmark datasets. As Mnist has multiple classes, we display the digit 0 and 1 classes in Fig. 1 (row 1) for ease of comparison. The Annthyroid anomaly dataset in Fig. 1 (row 2) distinguishes normal and anomaly samples. Fig. 1 demonstrates the key differences of the two LNL problems.

C1: Diversity of Anomalies. Comparing Fig. 1(a) to Fig. 1(f) we observe that for classification tasks, instances within a class tend to be similar to each other thus forming clusters [9], whereas anomalies can originate from a potentially infinite variety of distributions [5]. Therefore, the anomaly class lacks an identifiable structure. This causes many *marginal* samples to arise – i.e., subtle anomalies or obscure inliers that share characteristics with the opposite class and thus appear in overlapping portions of the feature space. In our noisy label context, such marginal samples are extra challenging to discern from clean samples as even anomalies with clean labels present no learnable structure to compare to.

C2: Clean Anomaly Scarcity. As shown in Fig. 1(f) and Tab. 2, anomaly samples are by definition rare and are thus significantly outnumbered by inliers; whereas classification tasks are often more balanced (Fig. 1(a)). Since networks tend to learn more quickly from the majority class than the minority class [24, 38], anomaly samples – even with clean labels – are harder to learn than inliers. This unfortunately hinders commonly deployed LNL metrics, such as loss [13, 36, 39, 41], to discern noisy samples; having assumed they are harder to learn than clean ones. This is illustrated by the clear separation of loss distributions in Fig. 1(b-c) for LNL-C compared to the overlapping distributions in Fig. 1(h) for LNL-AD.

C3: Unknown Disparate Noise Rates. The quality of initial pseudo-labels for anomaly datasets, as measured by the noise rate, can vary significantly (Tab. 2) and is thus difficult to determine without access to true labels. Further, as seen in Fig. 1(i-j) and Tab. 2, the noise rate is disparate between the inlier and anomaly classes, with the anomaly class containing an overwhelming ratio of noisy samples to clean samples. These factors combine to make LNL in the anomaly context particularly challenging as an accurate noise rate is often required for determining which and how many labels are clean [13, 36, 39, 41]. Even an accurate estimate of the overall rate will inevitably be quite inaccurate for at least one class.

State-of-the-Art and its Limitations. LNL techniques can be broadly categorized into sample selection versus label refurbishment. *Sample selection* methods, introduced for tackling LNL for *classification* [13, 39, 41], leverage the small loss observation to select a subset of samples with likely clean labels (aka, clean samples). Namely, mislabeled (noisy samples) are assumed to be harder to learn and thus to have a higher loss during training [10, 16, 19, 22]. This unfortunately no longer holds in the context of LNL-anomaly detection (Challenge C2). Although AutoOD [3]

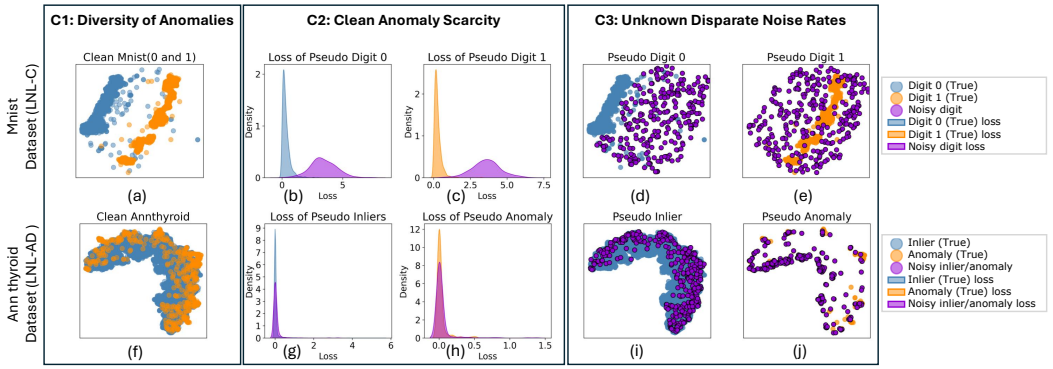


Fig. 1. Effects of pseudo-label noise in anomaly detection vs classification. Annthyroid pseudo-labels generated by an unsupervised Isolation Forest. Mnist pseudo labels generated by applying 20% synthetic random noise to ground truth labels per existing LNL-C works [13, 36, 39, 41]. Datasets visualized using t-SNE. Loss values recorded at training epoch 10.

recently applied sample selection for anomaly detection, it failed to consider the scarcity of clean anomalies (Challenge C2). As the loss metric fails to separate clean from noisy anomalies. Further, with the high initial label noise rate arising in the anomaly class, complicating safe sample selection (Challenge C3), AutoOD tends to predominantly select inliers and only a few easy anomalies.

In contrast, *label refurbishment* methods, again in the classification context [7, 33, 46], aim to identify and correct mislabeled samples using prediction probabilities. Recently, UADB [40] applied this approach to anomaly detection by assuming anomalies exhibit greater prediction variance than inliers. They use this variance signal to refurbish noisy samples. Unfortunately, due to the diversity of anomalies (Challenge C1), a large number of marginal samples exist that result in conflicting prediction probability signals during training, leading to false refurbishments.

Our Approach. To overcome the above challenges, we propose UNITY, the first method to combine sample selection and label refurbishment to learn from initial noisy pseudo-labels for anomaly detection. UNITY features 3 key innovations described below.

(1) UNITY's first innovation rests on an important observation that while peer networks (models with common architectures but different initialized weights) commonly utilized by LNL methods, [13, 36, 39, 41] tend to *agree* on predictions for "easy" training samples and *disagree* for "hard" samples. To use this for anomaly detection, we design UNITY's *Agree-to-Disagree* clean sample selection strategy (Sec. 3) which effectively resolves peer network disagreements while avoiding the invalidation of the loss as metric due to anomaly scarcity (Challenge C2). By jointly leveraging both network agreement and disagreement, UNITY selects a diverse set of easy and also marginal (hard) samples with clean labels that existing selection-based approaches would otherwise discard.

(2) Second, unlike existing LNL methods [13, 36, 39, 41] which assume the ground truth noise rate is given to them (Challenge C3), UNITY does not require this often unavailable knowledge. Instead, we design an *adaptive thresholding* methodology (Sec. 4) that leverages anomaly-aware statistics to separate the clean samples from noisy samples and achieves performance competitive to that of using the ground truth noise rates (Sec. 6.6).

(3) Third, rather than discarding the extremely hard-to-learn samples not selected by our Agree-to-Disagree method, we observe that these samples are disproportionately marginal and thus likely to capture the diversity of the data. We therefore develop an *anomaly-centric transformation network* CONTRASTCORR (Sec. 5) that uses contrastive learning to transform these samples into a new anomaly-aware embedding space that separates marginal clean from noisy samples. This

transformation overcomes the lack of learnable structure in the feature space (Challenge C1) and enables UNITY to reliably identify and refurbish mislabeled samples (Sec. 6.4).

By harnessing unique properties of anomalies, UNITY selects diverse samples with clean labels relevant for network training. At each training epoch, UNITY utilizes the refined set of selected and refurbished samples for jointly re-training the peer networks. These newly trained networks then predict improved pseudo-labels, jump-starting the next iteration of network training and refinement. This leads to a diverse and increasingly clean set of labels over time.

Our experiments on 10 benchmark anomaly detection datasets demonstrate that UNITY successfully decreases the noise rate by 2-7 fold compared to the initial pseudo-labels, while still including 68%-73% of samples for training (Sec. 6.4). This, in turn, leads to consistently improved anomaly detection, with UNITY outperforming the state-of-the-art by as much as 0.31 in F1 Score (Sec. 6.2).

Contributions. Our key contributions include the following

- Propose the *Agree-to-Disagree* sample selection strategy which integrates agreement and disagreement-based metrics on peer networks to overcome the limitations of loss alone (Challenge C2) to accurately estimate label cleanliness.
- Develop an anomaly-centric adaptive thresholding strategy that reliably divides likely clean inliers and anomalies from their noisy counterparts without requiring to be given the actual class noise rates (Challenge C3).
- Design CONTRASTCORR, a contrastive learning approach for leveraging *Agree-to-Disagree*'s clean samples to learn an embedding space that reveals informative marginal samples ready for refurbishment; otherwise difficult to discern (Challenge C1).
- Evaluate UNITY on ten benchmark anomaly detection datasets and demonstrate that UNITY consistently outperforms the state-of-the-art LNL techniques across a range of noise rates.

2 UNITY Overview

We first define our problem and then overview our proposed UNITY method. Please refer to Tab. 1 for the notations used in this work.

Notation	Description
$\mathbf{X}$	Given feature data
$\tilde{\mathbf{Y}}$	Given initial labels that contain noise
$\hat{\mathbf{Y}}$	Predicted labels
$\mathbf{Y}$	Ground-truth labels
$\hat{\mathbf{Y}}^*$	Refurbished label
$f(\cdot;\theta)$	Anomaly classifier with weights θ
p	Pair of samples
$\mathbf{X}^{\mathbf{p}}$	Set of pairs
$\hat{y}^c$	Contrastive learning label for a pair of samples
$\mathbf{X}^{\mathbf{s}}$	Samples selected as having clean labels
$\mathbf{X}^{\mathbf{r}}$	Samples with refurbished labels
$\mathbf{X}^{\mathbf{tr}}$	Samples to be used for training
$\mathbf{X}^{\mathbf{o}}$	Non-selected samples
$\mathbf{X}^{\mathbf{p}}$	Pairs of samples
ϵ	Noise Rate

Table 1. Table of notation.

2.1 Problem Definition

Given pseudo-labels generated by an unsupervised detector, the noise rate is defined below.

Definition 1. (Label Noise Rate of Pseudo Labels). Let $\mathbf{X} = \{x_i \in \mathbb{R}^d\}_{i=1}^N$ be the anomaly dataset with N samples and d features, $\mathbf{Y} = \{y_i \in [0,1]\}_{i=1}^N$ be $\mathbf{X}$'s ground truth labels where $y_i = 1$ means

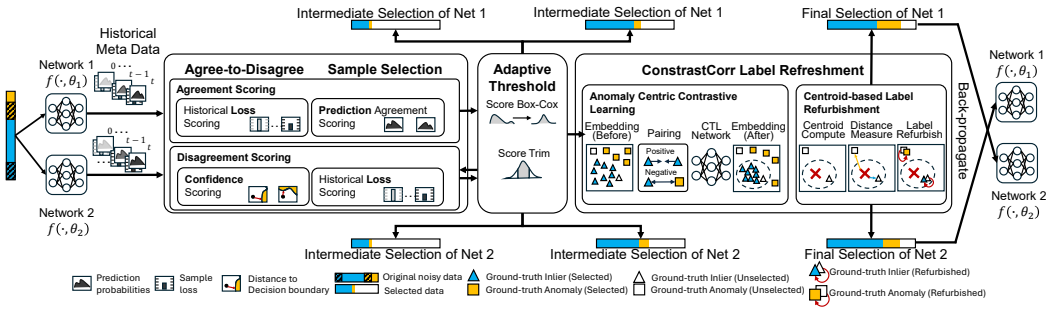


Fig. 2. UNITY Overview. Given the initial noisy data, each peer network collects historical meta data. Agree: calculates an agreement score for each sample using meta data. Adaptive Threshold: determines how many samples to select. Disagree: remaining samples receive new disagreement scores with the adaptive threshold applied to set a new threshold. CONTRASTCORR: refurbishes labels for samples not selected by Agree-to-Disagree. Peer networks are updated using cross-training.

that x_i is an anomaly and $y_i = 0$ means that x_i is an inlier. Also, let $\tilde{\mathbf{Y}} = \{\tilde{y}_i \in [0,1]\}_{i=1}^{N}$ be $\mathbf{X}$'s pseudo-labels. The label noise rate ϵ of $\tilde{\mathbf{Y}}$ is defined as the ratio between the number of wrong labels in $\tilde{\mathbf{Y}}$ versus the total number of samples N. Formally, ϵ is defined as:

$$\epsilon = \frac{\sum_{i=1}^{N} \mathbf{1}_{\{\tilde{y}_i \neq y_i\}}}{N} \tag{1}$$

where, $\mathbf{1}_{\{\tilde{y}_i \neq y_i\}}$ is the indicator function.

Specifically, we define the label noise rates ϵ_a and ϵ_i in the anomaly and inlier classes, respectively, as follows:

$$\epsilon_a = \frac{\sum_{i=1}^{N} \mathbf{1}_{\{y_i=1\}} \cdot \mathbf{1}_{\{\tilde{y}_i \neq y_i\}}}{\sum_{i=1}^{N} \mathbf{1}_{\{y_i=1\}}} \tag{2}$$

$$\epsilon_i = \frac{\sum_{i=1}^{N} \mathbf{1}_{\{y_i=0\}} \cdot \mathbf{1}_{\{\tilde{y}_i \neq y_i\}}}{\sum_{i=1}^{N} \mathbf{1}_{\{y_i=0\}}} \tag{3}$$

In practice, as seen in Fig. 1 and Tab. 2, the noise rate of the anomalies ϵ_a is much greater than that of the inliers ϵ_i, i.e. disparate noise rates. Using the noise rate definition, we define our problem.

Definition 2. (Anomaly Detection with Noisy Pseudo Labels). Let $\mathbf{X} = \{x_i \in \mathbb{R}^d\}_{i=1}^{N}$ be an anomaly dataset with N samples and d features, and $\tilde{\mathbf{Y}} = \{\tilde{y}_i \in [0,1]\}_{i=1}^{N}$ be the pseudo-labels generated by a basic unsupervised detector $h(\mathbf{X})$, with an unknown label noise rate ϵ. Our problem is to select a subset $\mathbf{X^{tr}} \subset \mathbf{X}$ of samples with **refined** pseudo labels $\hat{\mathbf{Y}}^{tr}$ such that the noise rate of the subset for each class $\hat{\epsilon}_a \ll \epsilon_a$ and $\hat{\epsilon}_i \ll \epsilon_i$. Further, an anomaly detector $f(\mathbf{X^{tr}}, \hat{\mathbf{Y}}^{\mathbf{tr}}; \theta)$ trained on $\mathbf{X^{tr}}$ and $\hat{\mathbf{Y}}^{tr}$ should achieve better performance on a given dataset, $\mathbf{X}$, than its counterpart $f(\mathbf{X}, \tilde{\mathbf{Y}}; \theta)$ trained on the same dataset $\mathbf{X}$ with its initial pseudo labels $\tilde{\mathbf{Y}}$, and the basic detector $h(\mathbf{X})$.

2.2 Overview of UNITY System

UNITY successfully learns from noisy labels for anomaly detection with 3 novel techniques: (1) *Agree-to-Disagree* sample selection strategy which selects samples with likely clean labels $\mathbf{X^s}$, (2) an *adaptive thresholding* technique that determines how many samples to select without requiring prior knowledge of the true noise rate, and (3) *refurbishment of labels in a transferred embedding space* which refurbishes noisy labels to have likely clean labels $\mathbf{X^r}$. Together these techniques

address all three challenges listed in Sec. 1 and thus produce a reliable set of sufficiently many clean samples, including marginal, needed for network training $\mathbf{X^{tr}} = \mathbf{X^{s}} \cup \mathbf{X^{r}}$.

Fig. 2 overviews UNITY composed of Agree-to-Disagree Sample Selector, Adaptive Threshold, and CONTRASTCORR Label Refurbishment. Inspired by Co-teaching [13], UNITY cross-trains two deep neural networks (peer networks), with the same architectures but different initialized weights, as seen in the first step of Fig. 2. These two networks communicate with each other to determine which samples are clean and should thus be used in the next iteration of training to update the weights of the other network.

During training, UNITY routes the samples to the Agree-to-Disagree sample selection component (Sec. 3). Here, UNITY identifies *clean easy*, and *clean marginal* samples. UNITY considers a sample clean if the two neural networks agree on its prediction and produce a small loss. However, since the instantaneous loss can fluctuate and becomes less informative in later epochs, we aggregate historical losses giving more weight to earlier observations.

After discovering the easy clean samples by agreement, UNITY leverages the signals hidden in the disagreement of the peer networks to identify the more challenging clean marginal samples – including more critically needed clean anomalies. The key idea is to leverage the distinct learning abilities of each network to select samples such that one network can successfully teach the correct prediction to the other network. To achieve this, UNITY combines the historical loss metric with the network's prediction confidence to identify samples that one network is confident in predicting while the other network is not.

Rather than assuming that the noise rate is known beforehand and thus hard-coding a fixed threshold on these scores to identify which labels are considered clean, UNITY's adaptive thresholding technique (Sec. 4) collects statistics on first the agreement and then disagreement scores to estimate their appropriate clean sample selection threshold with respect to both predicted inlier and anomaly labels. The estimated adaptive thresholds successfully separate clean samples from noisy ones without requiring access to the ground truth label noise rate.

Once the Agree-to-Disagree sample selection identifies the likely clean samples, the remaining samples with unknown label cleanliness are passed to the CONTRASTCORR Label Refurbishment component. Rather than discarding these samples as done by state-of-the-art methods, UNITY successfully refurbishes the labels of these marginal samples and merges them with the already selected samples to create an even more diverse training set (Sec. 5). Our unique approach is to leverage the clean samples selected by agree-to-disagree to *train a contrastive embedding network, that produces a new anomaly-centric embedding space resilient to label noise.* This anomaly-aware contrastive training pulls inlier samples in the embedding space close together, while anomalies are not pulled together as would have been done for classification but are instead simply pushed far from the inliers. This facilitates UNITY's ability to assign a newly corrected label to noisy samples while preserving labels of marginal clean samples, preventing false refurbishment. This overall UNITY process is summarized in Algo. 1.

3 Agree-to-Disagree Sample Selection

In this section, we describe UNITY's Agree-to-Disagree sample selection process which identifies likely clean samples. Intuitively, if two networks agree on a prediction, that prediction is likely to be correct. To leverage this, we introduce an Agreement-Historical Loss score w_{agree} (Sec. 3.1) which is used to select a likely clean sample set $\mathbf{X}_{\theta}^{\mathbf{agree}}$ through adaptive thresholding (Sec. 4). Here the historical loss serves to mitigate the effect of weak or instantaneous agreement to avoid selecting non-clean samples. Additionally, as the peer networks are unlikely to agree on a prediction for the more difficult yet highly informative marginal samples, we scrutinize the samples not in $\mathbf{X}_{\theta}^{\mathbf{agree}}$ via

Algorithm 1 UNITY: Overall Approach and Process Flow

Input: Networks $f(\cdot;\theta_1)$, $f(\cdot;\theta_2)$, $g(\cdot;\theta_3)$, dataset $\mathbf{X}$, initial noisy labels $\tilde{\mathbf{Y}}$, max epoch T, and warm-up epochs W
Output: De-noised Labels $\hat{\mathbf{Y}}_1$

for $t = 1$ **to** W **do**
  WARM-UP $f(\cdot;\theta_1)$ AND $f(\cdot;\theta_2)$ ON $\mathbf{X}$, $\tilde{\mathbf{Y}}$
for $t = W$ **to** T **do**
  // Agree-to-Disagree sample selection
  $\hat{\mathbf{Y}}_1, \mathbf{L}_1$ = FORWARDPASS $(f(\cdot;\theta_1), \tilde{\mathbf{Y}}, \mathbf{X})$ //Obtain prediction and loss of network 1
  $\hat{\mathbf{Y}}_2, \mathbf{L}_2$ = FORWARDPASS $(f(\cdot;\theta_2), \tilde{\mathbf{Y}}, \mathbf{X})$ //Obtain prediction and loss of network 2
  // Select samples with clean labels
  $\mathbf{X}_1^{\mathbf{s}}, \mathbf{X}_2^{\mathbf{s}}$ = AGREE-TO-DISAGREESAMPLESELECT$(\hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_2, \mathbf{L}_1, \mathbf{L}_2, \mathbf{X})$
  // ContrastCorr label refurbishment
  $\hat{\mathbf{Y}}_1^*, \hat{\mathbf{Y}}_2^*$ = CONTRASTCORRREFURBISH$(\mathbf{X}_1^{\mathbf{s}}, \mathbf{X}_2^{\mathbf{s}}, \hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_2, \mathbf{X}, g(\cdot;\theta_3))$
  // Replace the labels of non-selected samples with refurbished labels
  Overwrite $\hat{\mathbf{Y}}_1$ for non-selected samples with $\hat{\mathbf{Y}}_1^*$
  Overwrite $\hat{\mathbf{Y}}_2$ for non-selected samples with $\hat{\mathbf{Y}}_2^*$
  // Cross update networks
  $f(\cdot;\theta_1) \leftarrow$ NETWORKUPDATE$(f(\mathbf{X}_2^{\mathbf{s}};\theta_1), \hat{\mathbf{Y}}_2)$
  $f(\cdot;\theta_2) \leftarrow$ NETWORKUPDATE$(f(\mathbf{X}_1^{\mathbf{s}};\theta_2), \hat{\mathbf{Y}}_1)$
$\hat{\mathbf{Y}}_1 = f(\mathbf{X};\theta_1)$
return $\hat{\mathbf{Y}}_1$

a Disagreement-Historical Loss score $w_{disagree}$ (Sec. 3.2). Intuitively, this score uses the networks' confidences to estimate how likely each network is to teach the correct prediction for a sample to its peer and is also thresholded to produce a likely clean set $\mathbf{X}^{\mathbf{disagree}}$. Combined, these metrics capture a set of diverse highly likely clean samples, $\mathbf{X}_\theta^{\mathbf{s}} = \mathbf{X}_\theta^{\mathbf{agree}} \cup \mathbf{X}_\theta^{\mathbf{disagree}}$, that preserves the valuable marginal clean samples that SOTA selection methods often discard. This process is shown visually in Fig. 3.

3.1 Agreement-Historical Loss Scoring

To overcome the challenge of clean anomaly scarcity (Challenge C2), UNITY selects initial samples with likely clean labels by extending the popular small loss observation [3, 13, 16] to include peer network agreement. In essence, given the peer networks $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$, UNITY identifies samples as clean when the peer networks agree on their predictions for a sample and both generate a small loss.

Combining Historical Loss with Prediction Agreement. To integrate these concepts we aim to create a score that is large when the networks agree with low losses, and small otherwise. This corresponds to the blue and orange shaded regions of the feature space on the left of Fig. 3. For this, we calculate the **Agreement-Historical Loss** score $w_{agree}(x_i)$ of a sample x_i by combining the historical loss $w_{\mathcal{L}_c}(x_i)$ with the prediction agreement $w_D(x_i)$ of the two collaborating networks as expressed in Eq. 4.

$$w_{agree}(x_i) = \alpha \cdot w_D(x_i) + (1-\alpha) \cdot w_{\mathcal{L}_c}(x_i) \tag{4}$$

With **historical loss** $w_{\mathcal{L}_c}$ and **agreement** w_D defined below and in Eq. 5 and Eq. 7, respectively. Here α is a constant set to 0.5, giving equal preference to loss and agreement. After computing these

scores on all samples, UNITY selects the samples with maximal (and thus likely clean) agreement scores using adaptive thresholding.

While small loss for a sample x_i is a well-established sign for label cleanness in classification, the scarcity of clean anomalies (Challenge C2) leads the networks to quickly learn the majority inlier class, leaving clean anomalies with large losses similar to that of noisy samples. Further, the small loss signal at a given iteration of training can fluctuate randomly [48]. Therefore, relying solely on loss faces issues as stated below.

Remark 1. (Loss Loses its Separating Ability). *Given a large network $f(\cdot;\theta)$ and sample x_i, the training objective for classification is to minimize the loss produced by the network on the sample, $\min_\theta \ell\left(f(x_i;\theta),\tilde{y}\right)$, regardless of whether the sample is clean or noisy. Therefore as the training of $f(\cdot;\theta)$ progresses, the network will increasingly overfit on the pseudo labels, with clean inliers being overfit the fastest due to their overwhelming class majority. This results in the loss over time ceasing to meaningfully discriminate between clean anomalies and noisy inliers.*

Given this, we leverage the more informative historical loss instead of the instantaneous loss. Further, rather than providing larger weights to more recent observations as is typical, Rem. 1 leads us to prioritize the higher-quality loss signal from the *earlier* more informative epochs. To achieve this, our **historical loss score** $w_{\mathcal{L}_c}(x_i)$ uses the normalized Exponential Moving Average (EMA) of the historical loss trajectory of two peer networks $f(x_i;\theta_1)$ and $f(x_i;\theta_2)$ as in Eq. 5. To provide a larger score to small loss samples, we subtract the normalized historical losses from 1.

$$w_{\mathcal{L}_c}(x_i) = 1 - \left(\ell_t(f(x_i;\theta_1),\tilde{y}_i) + \ell_t(f(x_i;\theta_2),\tilde{y}_i)\right) \tag{5}$$

with ℓ_t being the EMA of binary cross-entropy loss below

$$\ell_t(f(x_i,\theta),\tilde{y}_i) = \begin{cases} (1-\beta)\cdot \ell_{BCE}(f(x_i;\theta),\hat{y}_i) + \\ \quad \beta \cdot \ell_{t-1}(f(x_i;\theta),\tilde{y}_i), & \text{if } t \geq 1 \\ \ell_{BCE}(f(x_i;\theta),\hat{y}_i), & \text{else} \end{cases} \tag{6}$$

Here t denotes the current training epoch. $\beta \in [0,1]$ a tuning parameter balancing the accumulated loss vs current loss. We set β to 0.9 to give more weight to earlier loss observations [48].

As illustrated earlier (Fig. 1), loss alone does not separate noisy anomalies from clean anomalies. We thus incorporate the networks' **prediction agreement** $w_D(x_i)$ of peer networks into our UNITY selection score based on the generated prediction probability distributions. To measure the symmetric similarity between predictions, UNITY uses the Jensen-Shannon (JS) divergence between the posteriors from each network's prediction for x_i as in Eq. 7 below. Since the JS divergence produces a small score between 0 and 1 for similar distributions, UNITY inverts the score by subtracting it by 1.

$$w_D(x_i) = 1 - JS\left(f(x_i;\theta_1)||f(x_i;\theta_2)\right) \tag{7}$$

where

$$\begin{aligned} JS\left(f(x_i;\theta_1)||f(x_i;\theta_2)\right) = \frac{1}{2} D_{KL}\left(f(x_i;\theta_1)||\frac{f(x_i;\theta_1)+f(x_i;\theta_2)}{2}\right) \\ + \frac{1}{2} D_{KL}\left(f(x_i;\theta_2))||\frac{f(x_i;\theta_1)+f(x_i;\theta_2)}{2}\right) \end{aligned}$$

With D_{KL} computing the classical Kullback–Leibler (KL) divergence that measures the relative entropy of the two distributions. A larger prediction agreement score means that two networks have similar predictions and thus are in agreement for a sample, and vice versa.

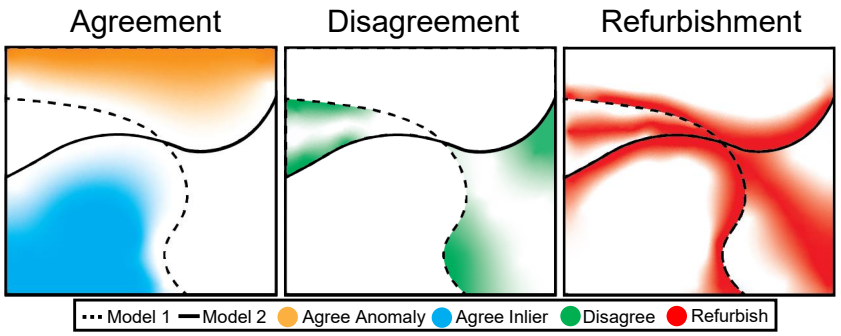


Fig. 3. Selection regions in feature space by components.

3.2 Disagreement-Historical Loss Scoring

While the above method securely selects samples easier to learn and far from the decision boundary for both networks, it struggles with the harder-to-learn clean marginal samples closer to the decision boundaries. Despite being risky to select, these marginal samples harbor critical information for improved anomaly detection performance. Thus, we aim to recover this information by the design of the Disagreement Resolver which acts on the samples *not* selected by Agreement. The region of the feature space that the Disagreement Resolver aims to select samples from is highlighted in green in the middle chart of Fig. 3.

Intuitively, we measure the confidence of the peer network's predictions based on the proximity of the sample to each decision boundary and use the more confident network to teach the less confident network the correct label. However, relying solely on a network's confidence can be risky [12, 27] due to the potential of false, overconfident predictions. To mitigate this risk, UNITY combines the confidence score with the network historical loss to create $w_{disagree}$, a Disagreement-Historical Loss score as follows.

Combined Disagreement-Loss Scoring. At each epoch, for each sample not selected as clean by the agreement selector (Sec. 3.1), we calculate the **Disagreement-Historical Loss** score.

$$w_{disagree}(x_i, \theta) = \alpha \cdot w_C(x_i, \theta) + (1 - \alpha) w_{\mathcal{L}_s}(x_i, \theta) \tag{8}$$

With w_C being the confidence for network $f(\cdot;\theta)$ calculated by Eq. 9, $w_{\mathcal{L}_s}$ the EMA binary cross entropy loss calculated by Eq. 10, and α the same scoring parameter as in Eq. 4.

For the above, we now define the notion of network confidence corresponding to the difference between each network's prediction and the decision boundary.

Definition 3. (Network Confidence). *Given a sample x_i, and a Network $f(\cdot;\theta)$, let the network confidence be the absolute distance between the network's posterior probability for the given sample, $f(x_i;\theta)$, and the decision boundary threshold σ.*

More precisely, we measure the **network confidence** of networks $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$ using their posterior predictions – i.e., the probability of a sample being an anomaly – as in Eq. 9.

$$\begin{aligned} w_C(x_i, \theta_1) &= |f(x_i;\theta_1) - \sigma| - |f(x_i, \theta_2) - \sigma| \\ w_C(x_i, \theta_2) &= |f(x_i;\theta_2) - \sigma| - |f(x_i, \theta_1) - \sigma| \end{aligned} \tag{9}$$

With σ representing the decision boundary (probability 0.5). Unlike the Agreement score, each network calculates an independent confidence score for each sample and maintains its own set of clean samples. A network is deemed the teacher network for a sample x_i if the network is confident in its prediction while the other is not.

Lastly, since networks can be overly confident in a prediction, even when the prediction is incorrect [12, 27], we enhance this process by penalizing samples with large losses. Similar to above (Sec. 3.1), to leverage a more informative loss and avoid fluctuations from the instantaneous loss of a sample, we utilize the EMA-smoothed **historical cross-entropy loss.** However, since each peer network now selects its own set of confident samples for teaching the other network, each calculates its own binary cross-entropy loss:

$$w_{\mathcal{L}_s}(x_i, \theta) = 1 - \ell_t(f(x_i; \theta), \tilde{y}_i) \tag{10}$$

with ℓ_t the EMA of the binary cross-entropy loss calculated below

$$\ell_t(f(x_i, \theta), \tilde{y}_i) = \begin{cases} (1-\beta) \cdot \ell_{BCE}(f(x_i; \theta), \hat{y}_i) + \\ \quad \beta \cdot \ell_{t-1}(f(x_i; \theta), \tilde{y}_i), & \text{if } t \geq 1 \\ \ell_{BCE}(f(x_i; \theta), \hat{y}_i), & \text{else} \end{cases} \tag{11}$$

where t is the current epoch of training, and $\beta \in [0, 1]$ is a tuning parameter set to 0.9. To prioritize small loss samples the historical losses are again normalized and inverted by subtracting from 1.

3.3 Agree-to-Disagree Algorithm

Agree-to-Disagree Sample Selection Algorithm. We pull together the complete Agree-to-Disagree sample selection strategy in Algo. 2. Given a dataset $\mathbf{X}$ and the loss and predictions generated by the networks $f(\cdot; \theta_1)$ and $f(\cdot; \theta_2)$, Algo. 2 selects subsets $\mathbf{X}_1^s$ and $\mathbf{X}_2^s$ out of $\mathbf{X}$ for the networks, respectively. Lines 1-3 first generate the selected subsets by the Loss-Prediction network Agreement (Sec. 3.1), using an estimated threshold on the computed agreement scores described in Sec. 4. Lines 4-7 select additional samples from the so-far unselected samples utilizing the Disagreement Resolver (Sec. 3.2). Lines 8-9 union the subsets selected by agreement and disagreement respectively for each network.

Complexity Analysis. The complexity of Algo. 2 is bounded by the complexity of agreement and disagreement selections. Let $|\mathbf{X}|$ be the size of the dataset. The complexity of agreement selection is $O(|\mathbf{X}|)$ since both its major components, EMA and KL Divergence, have linear computation costs of $O(|\mathbf{X}|)$. Similarly, the complexity of disagreement selection is $O(|\mathbf{X}|)$ since both network-confidence and classification loss scoring have computation costs of $O(|\mathbf{X}|)$. Lastly, as shown in Sec. 4, threshold estimation is also linear.

Algorithm 2 AGREE-TO-DISAGREESAMPLESELECT

Input: Dataset $\mathbf{X}$, predictions $\hat{\mathbf{Y}}_1$ and loss $\mathbf{L}_1$ of network $f(\cdot; \theta_1)$, and predictions $\hat{\mathbf{Y}}_2$ and loss $\mathbf{L}_2$ of network $f(\cdot; \theta_2)$

Output: Selected subsets of clean samples $\mathbf{X}_1^s$ and $\mathbf{X}_2^s$

1: $\mathcal{W}_{agree} = \text{AGREEMENTSCORE}(\hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_1, \mathbf{L}_1, \mathbf{L}_2)$; //by Eq. 4
2: $\delta_{agree} = \text{ADAPTIVETHRESHOLD}(\mathcal{W}_{agree})$; //by Eq. 12
3: $\mathbf{X}_1^s, \mathbf{X}_2^s = \{x \in \mathbf{X} \mid w_{agree}(x) \geq \delta_{\mathcal{W}_{agree}}\}$;
4: $\mathcal{W}_{disagree_1} = \text{DISAGREEMENTSCORE}(\mathbf{X} \setminus \mathbf{X}_1^s, \hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_2, \mathbf{L}_1, \mathbf{L}_2))$; //by Eq. 8
5: $\mathcal{W}_{disagree_2} = \text{DISAGREEMENTSCORE}(\mathbf{X} \setminus \mathbf{X}_2^s, \hat{\mathbf{Y}}_1, \hat{\mathbf{Y}}_2, \mathbf{L}_1, \mathbf{L}_2))$; //by Eq. 8
6: $\delta_{disagree_1} = \text{ADAPTIVETHRESHOLD}(\mathcal{W}_{disagree_1})$; //by Eq. 12
7: $\delta_{disagree_2} = \text{ADAPTIVETHRESHOLD}(\mathcal{W}_{disagree_2})$; //by Eq. 12
8: $\mathbf{X}_1^s = \mathbf{X}_1^s \cup \{x \in \mathbf{X} \setminus \mathbf{X}_1^s \mid w_{disagree_1}(x) \geq \delta_{\mathcal{W}_{disagree_1}}\}$;
9: $\mathbf{X}_2^s = \mathbf{X}_2^s \cup \{x \in \mathbf{X} \setminus \mathbf{X}_2^s \mid w_{disagree_2}(x) \geq \delta_{\mathcal{W}_{disagree_2}}\}$;
10: **return** $\mathbf{X}_1^s, \mathbf{X}_2^s$

4 Anomaly-Adaptive Threshold

Once the samples are assigned sampling scores by Agreement (Sec. 3.1) or Disagreement (Sec. 3.2), UNITY must determine how many samples to select as having clean labels. The common approach is to set a hard-coded threshold based on a-priori knowledge of the ground truth noise rate [13, 36, 39, 41]. However, in practice, it is difficult to estimate the true noise rate when the ground truth labels are unknown. Worse yet, this approach assumes that the noise rate is equal across classes – an assumption that does not hold for anomaly detection (Challenge C3).

Since samples with larger sampling scores are more likely to have clean labels, a sample x_i will likely have a clean label $y_i = \hat{y}_i$ if either $w_{agree}(x_i)$ or $w_{disagree}(x_i, \theta)$ fall in the upper tail of the sample score distributions. Our first thought would thus be to leverage batch statistics, such as mean and standard deviation, as done in the literature [31] to determine the threshold. Unfortunately, this technique requires the sampling score distributions to be approximately normal to prevent the calculated threshold from being too small or too large and thereby selecting too many or too few samples. This assumption fails however when applied to anomaly detection because inliers are easier for networks to learn — resulting in smaller losses and higher confidence scores. This skews the sampling score distributions. Further, since anomalies lack clear structure (Challenge C1), it is not guaranteed nor even likely that the losses and confidences will be normally distributed.

To overcome these issues, we propose to adopt a two-step approach. First, we transform the sampling score distributions to be approximately normal. Thereafter, we exclude the extreme values [1] often present in anomaly detection that can bias the batch statistics. To transform the sample score distributions, UNITY leverages the Box-Cox transformation [2], shown in Eq. 13, as it can handle a wide range of transformations. Further, to exclude extreme values, UNITY trims the transformed sampling scores by excluding the top and bottom 5% of samples with the largest and smallest sampling scores from the threshold calculation.

The transformed-trimmed sample scores are then exploited to calculate the unsupervised threshold by adding the standard deviation of the transformed-trimmed scores to the mean of the transformed-trimmed scores, as shown in Eq. 12. Since inliers and anomalies have different noise rates (Challenge C3) and anomalies are known to have higher losses [3], UNITY splits the sampling scores into predicted inlier and predicted anomaly sets. Each set then calculates its own respective threshold. Any samples in the corresponding predicted sets with a sampling score greater than the calculated thresholds are deemed to have clean labels. They are thus selected as samples for the next round of training (Eq. 14).

$$\delta_\tau = \mu_k + \varphi \left(\sqrt{\frac{1}{n - (2 \cdot k)} \sum_{j=k+1}^{n-k} \left(w'_\tau(x_j) - \mu_k \right)^2} \right) \tag{12}$$

where $\mu_k = \frac{1}{n-(2 \cdot k)} \sum_{i=k+1}^{n-k} w'_\tau(x_i)$ is the trimmed mean, $k = n \cdot 0.05$ is the amount of samples to remove from both tails, $w'_\tau(x_i)$ is the Box-Cox [2] transformed sampling score τ given to sample x_i per Eq. 13, where λ is the automatically calculated maximum likelihood estimation, n is the number of samples, and φ is a scoring parameter that is set to 1 for the Agreement scoring and 1.5 for the Disagreement scoring such that agreement selects slightly more samples than disagreement.

$$w'_\tau(x_i) = \begin{cases} \frac{w_\tau(x_i)^\lambda - 1}{\lambda} & \text{if } \lambda \neq 0 \\ \log(w_\tau(x_i)) & \text{if } \lambda = 0 \end{cases} \tag{13}$$

The final selected sets are then computed at the union of the selected clean samples from agreement and disagreement as follows.

$$\mathbf{X}^{\mathbf{s}}_\theta = \mathbf{X}^{\mathbf{s}}_\theta \cup \{ x \in X \mid w'_\tau(x) \geq \delta_\tau \} \tag{14}$$

Anomaly-Adaptive Threshold Algorithm. We insert this adaptive threshold solution into our Agree-to-Disagree strategy in Algo. 2. Once each sample is given an Agreement-Historical Loss score (Sec. 3.1), the adaptive threshold is calculated on line 2. Samples with a score greater than the threshold are selected as likely being clean on line 3. Samples not selected are assigned our new Disagreement-Historical Loss scores (Sec. 3.2). The process repeats on lines 4-7.

Complexity Analysis. The complexity of the Anomaly-Adaptive Threshold Algorithm is bounded by the computation of Box-Cox transformation, which has a complexity of $O(|\mathbf{X}|)$[2].

5 ContrastCorr Label Refurbishment

Although the Agree-to-Disagree selection process accurately selects clean samples (Sec. 3), it largely excludes anomaly samples – despite those being crucial for supervised network training [5, 35]. This exclusion occurs due to the high initial noise rate among the anomaly class (Challenge C3), resulting in few anomaly samples having clean initial labels. To incorporate the valuable signal from these ignored anomaly samples, Unity seeks to refurbish as many of the noisy labels as safely as possible. The refurbished samples, $\mathbf{X^r}$, are then combined with the selected clean samples for network training $\mathbf{X^{tr}} = \mathbf{X^s} \cup \mathbf{X^r}$. Unfortunately, refurbishment is difficult because the samples remaining in the "non-selected" set, $\mathbf{X^o} = \mathbf{X} \setminus \mathbf{X^s}$ consist of noisy and hard-to-learn clean samples near the confidence boundaries of the peer networks, shown by red areas in the feature space of Fig. 3. Further, the diversity of anomalies (Challenge C1), makes it challenging to refurbish their labels using traditional label refurbishment techniques.

To overcome these issues, Unity introduces an anomaly-aware strategy, called ContrastCorr, for transforming the feature space into an anomaly-revealing embedding space (Sec. 5.1). For this, ContrastCorr leverages the samples from the high-quality clean label set $\mathbf{X^s}$ selected by the prior Agree-to-Disagree component. Instead of attempting to learn the hard-to-characterize structure of anomalies, this embedding space instead achieves greater separation between inliers and anomalies by forcing inliers to cluster together while pushing anomalies away using a one-directional approach. Leveraging this new embedding space, Unity calculates a new sampling score, w_{CTL}, based on the distance of each non-selected sample to the centroid of the embedded inlier cluster. This allows Unity to assign the correct labels to the non-selected set $\mathbf{X^o}$ either by changing the label or by keeping the original label if it is already correct (Sec. 5.2). These refurbished samples are then combined with the clean samples selected by Agree-to-Disagree $\mathbf{X^s}$ to form the revised set for the next round of network training.

5.1 Anomaly Centric Contrastive Learning

The goal of contrastive learning is to learn a new embedding space that pulls the embeddings of similar samples (positive pairs) closer together while pushing dissimilar samples (negative pairs) apart [18]. Adopting this principle to our problem setting, Unity leverages the selected clean sample set of both networks $\mathbf{X_1^s}$ and $\mathbf{X_2^s}$ to construct a contrastive learning training dataset for anomaly detection X^p. As described in Def. 4, instead of creating positive pairs of our two classes as (inlier, inlier) and (anomaly, anomaly), and forcing the embedding network to learn the difficult anomaly embedding region, ContrastCorr only constructs (inlier, inlier) positive pairs. As for the negative pairs, we work with (inlier, anomaly) pairs. Overall, we create $2\left|\{x_i \in \mathbf{X_1^s} | \hat{y}_i = 0\} \cup \{x_i \in \mathbf{X_2^s} | \hat{y}_i = 0\}\right|$ pairs p as in Eqs. 15 and 16. The embedding network $g(\cdot; \theta_3)$ is then trained on the pairs p using the ContrastCorr loss function in Def. 5.

Definition 4. (Contrastive Training Pairs). *Given two samples $x_i, x_j \in \mathbf{X^s}$ selected as having clean labels by the Agree-to-Disagree component, they are considered a positive pair iff both predicted labels are inlier $\hat{y}_i = 0, \hat{y}_j = 0$. Otherwise, if one sample is predicted anomaly $\hat{y}_i = 1, \hat{y}_j = 0$ or*

$\hat{y}_i = 0, \hat{y}_j = 1$, *they are a negative pair. The sets of positive and negative pairs with contrastive labels* $\hat{y}_p^c = 0$ *and* $\hat{y}_p^c = 1$ *respectively are:*

$$\mathbf{X}^{\mathbf{p}}_{\mathbf{pos}} = \left\{(x_i, x_j) | \hat{y}_i = \hat{y}_j = 0, i \neq j; \forall (x_i, x_j) \in \left(\mathbf{X}^{\mathbf{s}}_1 \cup \mathbf{X}^{\mathbf{s}}_2\right)^2\right\} \tag{15}$$

$$\mathbf{X}^{\mathbf{p}}_{\mathbf{neg}} = \left\{(x_i, x_j) | \hat{y}_i = 0 \wedge \hat{y}_j = 1; \forall (x_i, x_j) \in \left(\mathbf{X}^{\mathbf{s}}_1 \cup \mathbf{X}^{\mathbf{s}}_2\right)^2\right\} \tag{16}$$

Definition 5. (ContrastCorr Loss Function). *To train the embedding network* $g(\cdot; \theta_3)$ *to map a sample* x_i *to an embedding space* $g(x_i; \theta_3)$ *where the positive pairs are pulled together while the negative pairs are pushed away,* Unity *uses the contrastive loss in Eq. 17. This loss updates the network* $g(\cdot; \theta_3)$ *at each epoch.*

$$\begin{aligned} \ell_{CTL} = \frac{1}{|\mathbf{X}^{\mathbf{p}}|} \sum_{p=1}^{|\mathbf{X}^{\mathbf{p}}|} \Bigg(& (1 - \hat{y}_p^c) \cdot \left\| g(x_p^i; \theta_3) - g(x_p^j; \theta_3) \right\|^2 \\ & + \hat{y}_p^c \cdot \max\left(0, m - \left\| g(x_p^i; \theta_3) - g(x_p^j; \theta_3) \right\|\right)^2 \Bigg) \end{aligned} \tag{17}$$

where $|\mathbf{X}^{\mathbf{p}}|$ *is the number of pairs,* $\hat{y}_p^c$ *is the contrastive label for the pair* p, $\|\cdot\|$ *the Euclidean distance, and* m *a margin hyperparameter set to 1. Here, the first term, activated by positive pairs,* $\hat{y}_p^c = 0$ *, minimizes the distances between the pair of samples and the second term, activated by negative pairs,* $\hat{y}_p^c = 1$*, encourages the pair of samples to be pushed apart by a margin of* m*.*

Once the embedding network, $g(\cdot; \theta_3)$, is trained at the current epoch, the new anomaly-centric embedding space is used to assign improved labels to the remaining non-selected samples X^o.

5.2 ContrastCorr Label Refurbishment

Given our Agree-to-Disagree identifies a large set of clean predicted inliers and some clean anomalies, the newly learned embedding space will pull the set of predicted inlier samples into a tight singular inlier cluster while pushing anomalies away. ContrastCorr then calculates an inlier centroid, seen in Eq. 18, representative of the average location of inliers in the embedding space.

$$C_{in} = \frac{1}{|\mathbf{X}^{\mathbf{in}}|} \sum_{i=1}^{|\mathbf{X}^{\mathbf{in}}|} g(x_i; \theta_3) \tag{18}$$

Where $\mathbf{X}^{\mathbf{in}} = \{x_i \in \mathbf{X}^{\mathbf{s}}_1 | \hat{y}_i = 0\} \cup \{x_i \in \mathbf{X}^{\mathbf{s}}_2 | \hat{y}_i = 0\}$.

Thereafter, Unity passes the remaining non-selected samples $\mathbf{X^o}$ to $g(\cdot; \theta_3)$ to compute their corresponding embeddings. Unity then calculates the distances from the embeddings to the likely clean inlier centroid C_{in} shown in Eq. 19. Intuitively, the distances of the non-selected samples to the centroid of the likely inlier samples are treated as an anomaly score. That is, samples with large distances are likely to be ground truth anomalies and those with small distances are likely to be ground truth inliers. Unity leverages the noisy anomaly ratio to calculate the distance threshold to distinguish anomalies from inliers.

$$w_{CTL}(x_i) = \| g\left(x_i; \theta_3\right) - C_{in} \|_2 \tag{19}$$

The most confident, based on the distance to the inlier centroid, non-selected samples $\mathbf{X^r}$ with their refurbished labels $\hat{\mathbf{Y}}^*$ are then combined with the selected samples $\mathbf{X^s}$ to create the current epoch's training set, $\mathbf{X^{tr}}$. The training set is then used to cross update the networks $f(\cdot; \theta_1)$ and $f(\cdot; \theta_2)$.

Algorithm 3 ContrastCorrLabelRefurbish

Input: Dataset $\mathbf{X}$, selected subsets of clean samples $\mathbf{X}_1^{\mathbf{s}}$ and $\mathbf{X}_2^{\mathbf{s}}$ yielded from Algorithm 2, predicted labels $\hat{\mathbf{Y}}_1$ and $\hat{\mathbf{Y}}_2$, and model $g(\cdot;\theta_3)$
Output: Refurbishing labels belonging to $\hat{\mathbf{Y}}_1$ and $\hat{\mathbf{Y}}_2$

1: $\mathbf{X}^{\mathbf{p}} = \{(x_i, x_j) | \hat{y}_i = 0 \wedge \hat{y}_j = 0, i \neq j; \forall (x_i, x_j) \in \mathbf{X}_1^{\mathbf{s}} \times \mathbf{X}_2^{\mathbf{s}}\}$
2: $\mathbf{X}^{\mathbf{p}} = \mathbf{X}^{\mathbf{p}} \cup \{(x_i, x_j) | \hat{y}_i = 0 \wedge \hat{y}_j = 1, i \neq j, \forall (x_i, x_j) \in \mathbf{X}_1^{\mathbf{s}} \times \mathbf{X}_2^{\mathbf{s}}\}$
3: $\ell_{CTL} = \textsc{Forward}(\mathbf{X}^{\mathbf{p}}, g(\cdot;\theta_3))$ //by Eq. 17
4: $\theta_3 = \textsc{BackProp}(\theta_3, \ell_{CTL})$
5: **Obtain** centroid C_{in} //by Eq. 18
6: $\mathbf{X}_1^{\mathbf{o}} = \mathbf{X} \setminus \mathbf{X}_1^{\mathbf{s}}$
7: $\mathbf{X}_2^{\mathbf{o}} = \mathbf{X} \setminus \mathbf{X}_2^{\mathbf{s}}$
8: $\mathcal{W}_{CTL_1} = \textsc{ContrastCorrScore}(X_1^o, C_{in})$ //by Eq. 19
9: $\mathcal{W}_{CTL_2} = \textsc{ContrastCorrScore}(X_2^o, C_{in})$ //by Eq. 19
10: **Obtain** $\hat{\mathbf{Y}}_1^*$ from $\mathcal{W}_{CTL_1}$
11: **Obtain** $\hat{\mathbf{Y}}_2^*$ from $\mathcal{W}_{CTL_2}$
12: **return** $\hat{\mathbf{Y}}_1^*, \hat{\mathbf{Y}}_2^*$

5.3 ContrastCorr Algorithm

ContrastCorr Label Refurbishment Algorithm. Given dataset $\mathbf{X}$, selected subsets $\mathbf{X}_1^{\mathbf{s}}$ and $\mathbf{X}_2^{\mathbf{s}}$ returned by Algo. 2 and predicted labels, Algo. 3 refurbishes the labels of the unselected subset $\mathbf{X} \setminus \mathbf{X}_1^{\mathbf{s}}$ and $\mathbf{X} \setminus \mathbf{X}_2^{\mathbf{s}}$. Specifically, Lines 1-2 construct the positive and negative pairs as per Eqs. 15 and 16. Lines 3-4 compute the contrastive loss defined in Eq. 17 and update the network. With the contrastive network, Lines 5-9 first compute the centroid of the inliers defined in Eq. 18 and then calculate the ContrastCorr scores defined in Eq. 19. Lines 10-11 refurbish the labels using the ContrastCorr scores, which are returned in Line 12.

Complexity Analysis. Let $|\mathbf{X}|$ be the size of the dataset, d be the embedding dimensionality, and $|\theta_3|$ be the number of weights of network $g(\cdot;\theta_3)$. The complexity of Algo. 3 is bounded by the total complexity of (1) the pair construction in Lines 1-2, (2) the forward and backward passes of the network in Lines 3-4, (3) the centroid computation in Line 5 and (4) the ContrastCorr score computation in Line 8-9. In particular, since we are doing non-replacement sampling, the complexity of the random pair construction is bounded by $O(|\mathbf{X}|)$. The complexity of the forward and backward passes of a fully connected network is bounded by $O(d \times |\mathbf{X}| \times |\theta_3|)$. The complexity of the mean centroid computation is $O(|\mathbf{X}| \times d)$ and the complexity of the ContrastCorr score computation is $O(|\mathbf{X}| \times d)$. Together the complexity of Algo. 3 is bounded by $O(d \times |\mathbf{X}| \times |\theta_3|)$.

6 Experimental Evaluation

6.1 Experimental Setup

Benchmark Datasets. To verify the effectiveness of Unity, we use 10 common anomaly detection benchmark datasets [11, 25, 26, 32] with varying sizes, dimensionality, and anomaly ratios. Tab. 2 shows descriptive statistics about each dataset. Following [3], the datasets are feature normalized with duplicates removed.

Initial Noisy Labels. To generate initial noisy labels, we use an Isolation Forest (IF) anomaly detector [23] due to its simplicity, efficiency, and minimal hyper-parameter tuning [14]. The noise rates ϵ_a and ϵ_i for the anomaly and inlier classes respectively for all datasets are in Tab. 2. The rates

Dataset	# Instances	Dim	Anomaly Ratio (%)	Noise Rate ϵ_a (%)	Noise Rate ϵ_i (%)
Annthyroid	7,062	6	7.6	69	6
Cardio	1,822	21	9.6	51	5
Landsat	6,435	36	20.7	74	19
Mammography	7,848	6	3.2	84	3
Mnist	7,603	100	9.2	72	7
Pageblocks	5,393	10	9.5	59	6
Pendigits	6,870	16	2.3	62	1
Thyroid	3,656	6	2.5	48	1
Wine	129	13	7.8	90	8
WPBC	198	33	23.7	85	26

Table 2. Statistics of benchmark anomaly detection datasets.

differ significantly with the anomaly label noise rate ϵ_a ranging from 48% to 90%, and the inlier noise rate ϵ_i ranging from 1% to 26%.

Comparative Methods. We compare UNITY against three groups of methods: (1) initial label generator IF, (2) SOTA for Learning with Noisy Labels specific for Anomaly Detection, (LNL-AD), and (3) SOTA for Learning with Noisy Labels for Classification (LNL-C).

Initial Label Generator IF. Isolation Forest [23], a popular unsupervised *anomaly detector*, that generates initial pseudo labels [44].

Learning from Noisy Labels - Anomaly Detection (LNL-AD). We consider the two most recent methods for this group.

- **AutoOD** [3]: SOTA for learning from noisy labels for *anomaly detection* that prunes samples with large loss.
- **UADB** [40]: A SOTA for learning from noisy labels for *anomaly detection* that leverages the prediction variance between 2 networks to *refurbish* noisy labels.

Learning from Noisy Labels - Classification (LNL-C). We consider the following five popular methods for this group.

- **Co-teaching** [13]: *Selects* small loss samples as having clean labels for training.
- **Co-teaching+** [41]: Improved Co-teaching variant *selects* small loss samples with disagreeing predictions for training.
- **JoCoR** [39]: *Selects* small loss samples with agreeing predictions.
- **SEAL** [7]: *Refurbishes* noise labels for training by exploiting network prediction probabilities.
- **SELFIE** [36]: *Selects* small loss samples and *refurbishes* noisy labels leveraging prediction probabilities for training.

To make the above LNL-C methods compatible for anomaly detection, we implement them based on the same network architecture as UNITY. Since most LNL-C methods [13, 36, 39, 41], except SEAL [7] require the label noise rate for selection or refurbishment, we provide them with the respective ground-truth label noise rates for both anomaly and inlier classes. At selection time, the samples are split into predicted inliers and anomalies with the true noise rate for the respective class used to make selections.

Implementation Details. All experiments are conducted on an A100 GPU using the PyTorch [30] deep learning framework. The two peer networks $f(\cdot;\theta_1)$ and $f(\cdot;\theta_2)$ and the embedding network $g(\cdot;\theta_3)$ in UNITY each consists of 3 hidden layers. We use stochastic gradient descent with momentum of 0.9 and learning rate of 0.01 for the two peer networks and 0.001 for the embedding network. UNITY is ran for 200 epochs where the first 15 epochs are a warm-up period. These hyper-parameter settings are independent across all datasets and are held the same for all experiments. If hyper-parameters need to be tweaked, the training loss of the noisy labels can be leveraged. Some

datasets require batch norm layers to properly fit the data, which can be determined by monitoring the loss of the networks on the noisy labels. For fair comparison, methods are run with and without batch norm layers and report the best result. All code is released on GitHub[1] for reproducibility.

Metrics. We adopt F-1 score, commonly used by anomaly detection as an evaluation metric due to its support for class-imbalanced data. Each method is ran for 10 random seeds, with the top 3 seeds selected. We report the mean and standard deviations of F-1 scores.

Method	Datasets									
	Annthyroid	Cardio	Landsat	Mammography	Mnist	Pageblocks	Pendigits	Thyroid	Wine	WPBC
IF (Base)	$0.34_{.01}$	$0.54_{.02}$	$0.25_{.00}$	$0.26_{.01}$	$0.33_{.02}$	$0.44_{.01}$	$0.39_{.04}$	$0.61_{.04}$	$0.20_{.00}$	$0.17_{.02}$
AutoOD	$0.48_{.04}$	$0.12_{.04}$	$0.18_{.04}$	$0.25_{.03}$	$0.44_{.04}$	$0.49_{.06}$	$0.17_{.03}$	$\underline{0.79_{.01}}$	$\underline{0.52_{.15}}$	$\underline{0.32_{.03}}$
UADB	$0.39_{.02}$	$0.61_{.04}$	$0.31_{.03}$	$0.29_{.01}$	$0.34_{.01}$	$0.25_{.18}$	$0.47_{.17}$	$0.69_{.03}$	$0.18_{.06}$	$0.17_{.02}$
Co-teaching	$0.66_{.07}$	$0.56_{.03}$	$0.35_{.02}$	$0.30_{.11}$	$0.34_{.07}$	$0.50_{.01}$	$0.45_{.02}$	$0.77_{.05}$	$0.37_{.12}$	$0.26_{.02}$
Co-teaching+	$0.40_{.04}$	$0.57_{.04}$	$\underline{0.38_{.02}}$	$\underline{0.31_{.10}}$	$0.34_{.02}$	$0.48_{.03}$	$0.41_{.05}$	$0.73_{.02}$	$0.23_{.32}$	$0.23_{.06}$
JoCoR	$0.42_{.03}$	$0.56_{.05}$	$0.37_{.01}$	$0.10_{.04}$	$0.33_{.06}$	$0.53_{.03}$	$\underline{0.51_{.12}}$	$0.68_{.01}$	$0.50_{.36}$	$0.28_{.04}$
SEAL	$\mathbf{0.79_{.00}}$	$0.62_{.02}$	$0.32_{.01}$	$0.26_{.02}$	$0.37_{.03}$	$0.50_{.01}$	$0.48_{.02}$	$0.73_{.01}$	$0.20_{.00}$	$0.32_{.02}$
SELFIE	$0.66_{.03}$	$\underline{0.64_{.01}}$	$0.37_{.01}$	$0.27_{.11}$	$\underline{0.46_{.00}}$	$\underline{0.58_{.01}}$	$0.48_{.05}$	$0.72_{.01}$	$0.11_{.01}$	$0.28_{.10}$
UNITY (ours)	$\underline{0.78_{.00}}$	$\mathbf{0.65_{.09}}$	$\mathbf{0.40_{.02}}$	$\mathbf{0.34_{.08}}$	$\mathbf{0.48_{.02}}$	$\mathbf{0.63_{.01}}$	$\mathbf{0.58_{.09}}$	$\mathbf{0.81_{.02}}$	$\mathbf{0.83_{.15}}$	$\mathbf{0.36_{.02}}$

Table 3. F-1 Scores of the comparative methods and UNITY. The best method per dataset is **bolded**, and the second best is underlined. UNITY outperforms the compared methods, achieving the greatest performance on 9 of the 10 datasets, and yielding second best on its only non-winning dataset.

6.2 Comparative Effectiveness Evaluation

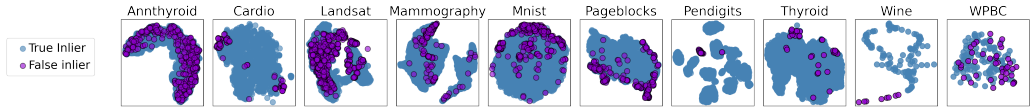
Tab. 3 shows the final anomaly detection results of all comparative methods on all 10 datasets. Overall, UNITY consistently outperforms all methods in 9 out of the 10 datasets, with the exception being Annthyroid, where UNITY is a close runner-up. Specifically, UNITY outperforms the SOTA LNL-AD methods, namely, AutoOD and UADB across all datasets by 0.02 to 0.31 points in F-1 score. UNITY also outperforms the LNL-C, methods on 9 out of the 10 datasets by 0.01 to 0.33 points in the F-1 score even though most of these methods are given the extra knowledge of the ground truth noise rates for both the inlier and anomaly classes, which our UNITY does not need. Further, we observe that compared to the initial label generator IF, only UNITY, Co-teaching, and Co-teaching+ provide a consistent improvement across all datasets with UNITY always showing the greater improvement.

By addressing the 3 critical challenges, UNITY outperforms the comparative methods. First, UNITY overcomes the effect the scarcity of clean anomalies has on sample selection metrics (Challenge C2) by leveraging agreement, disagreement, and historical loss metrics. Doing so allows Agree-to-Disagree (Sec. 3) to select clean samples including marginal samples that comparative methods discard. Then to include more anomalies, even in high noise settings (Challenge C3), UNITY learns a new anomaly-centric embedding space that adheres to the anomalies' lack of structure (Challenge C2) to accurately refurbish noisy labels.

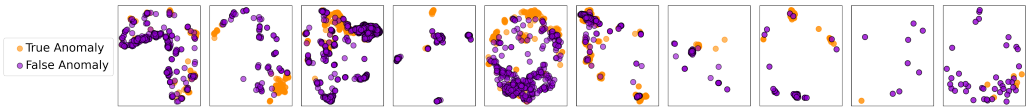
The initial label noise rate in the anomaly class ϵ_a in Tab. 2 has a substantial impact on overall anomaly detection performance. While most methods perform better on datasets with lower ϵ_a, Thyroid (48%), Cardio (51%), and Pageblocks (59%), they suffer severe performance degradation on datasets with higher ϵ_a, Landsat (74%), Mammography (84%), WPBC (85%), and Wine (90%). Despite this, UNITY consistently achieves the best performance. It achieves the highest global F-1 score of 0.833 on the Wine dataset with the highest ϵ_a of 90%. This performance can be attributed to UNITY's ability to use the signal from marginal samples to refurbish samples with noisy labels,

[1]https://github.com/dhofmann34/Unity

as discussed above. We further evaluate UNITY under different label noise rates ϵ_a in the anomaly class in Sec. 6.7.



(a) Pseudo inliers: true inliers vs. false inliers (true anomalies)



(b) Pseudo anomalies: true anomalies vs. false anomalies (true inliers)

Fig. 4. Pseudo inliers (First Row) and Pseudo anomalies (second row) colored by IF's predicted labels.

6.3 Case Studies using Benchmark Datasets

To demonstrate the challenges marginal samples create we provide Fig. 4. This figure shows the t-SNE visualization of each benchmark dataset split by their initial pseudo label with inliers in the top row and anomalies in the bottom row. The hue of the points represents the quality of the initial predicted labels by the IF detector. In the top row, the figure reveals that among the pseudo inliers, many false inliers fall in overlapping regions of true inliers. These are marginal samples as they are ground truth anomalies that exhibit similar characteristics as inliers, making them difficult to select or refurbish. Further, the bottom row of Fig. 4, shows that in addition to marginal samples, the pseudo anomalies contain many false anomalies making sample selection-only methods discard a large amount of anomaly samples.

6.4 Quality of Selected Samples by UNITY Stages

Fig. 5 depicts the selection quality of each UNITY stage. Each stage progressively selects a greater proportion of clean samples for network training. In total, approx. 68% to 73% of original samples are included in the training process. As expected, UNITY's Agree (Sec. 3.1) stage selects easier samples resulting in less mistakes. UNITY's Disagree selection (Sec. 3.2) targets the marginal samples with clean labels – adding additional samples to the training set. Although this selection is more risky and includes more noisy samples, the beneficial signal from the marginal samples outweighs the negative impact of a few noisy samples. Finally, UNITY's CONTRASTCORR refurbishes the labels of the samples not selected by UNITY's Agree-to-Disagree. In doing so, UNITY utilizes the majority of the dataset to update its peer networks, improving the downstream anomaly detection performance. By selecting less noisy samples, UNITY avoids conflicting signals during training which in turn improves anomaly detection.

Fig. 6 shows the reduction in the noise rate from the original noisy labels to the final set of selected and refurbished samples. The noise rate for anomalies decreased significantly, approximately 2 to 7 folds across all datasets. Although the noise rate is already relatively low for inliers, UNITY further reduces it by 1.5 to 4 fold.

6.5 Ablation Study of UNITY Components

In this study, we explore the contribution of each component of UNITY. We compare UNITY to its three variants: (1) **UNITY-A** which adopts UNITY's Agreement sample selection (Sec. 3.1) only,

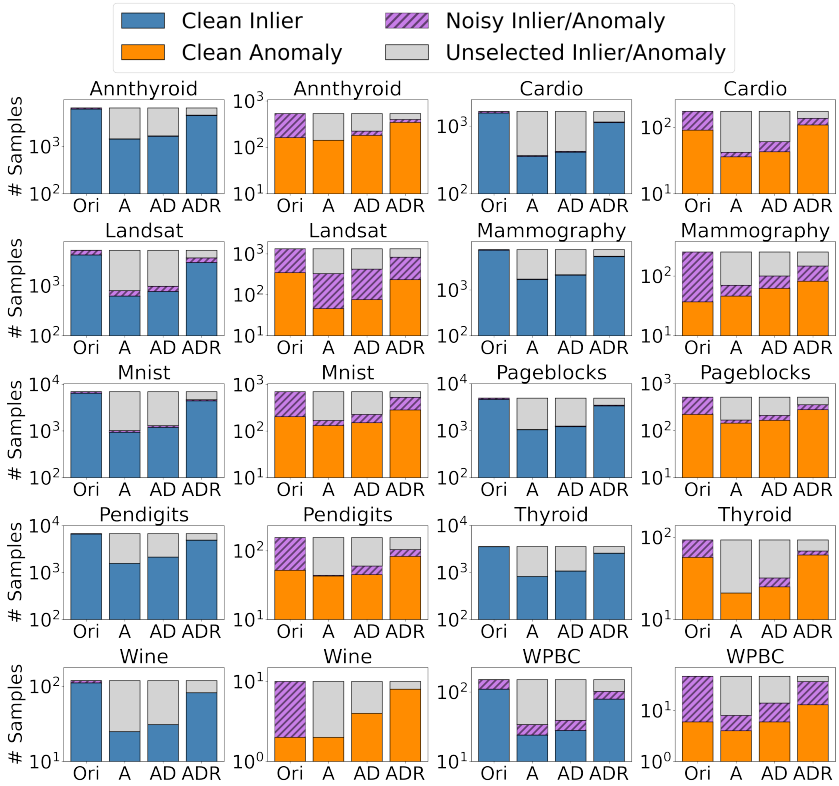


Fig. 5. Number of types of samples selected by UNITY stages (Epoch 200). Different types include clean inliers (blue) clean anomalies (orange) Noisy samples (purple) and unselected samples (gray). Stages include selection set after agreement (A), agreement and disagreement (AD), and all three stages (ADR).

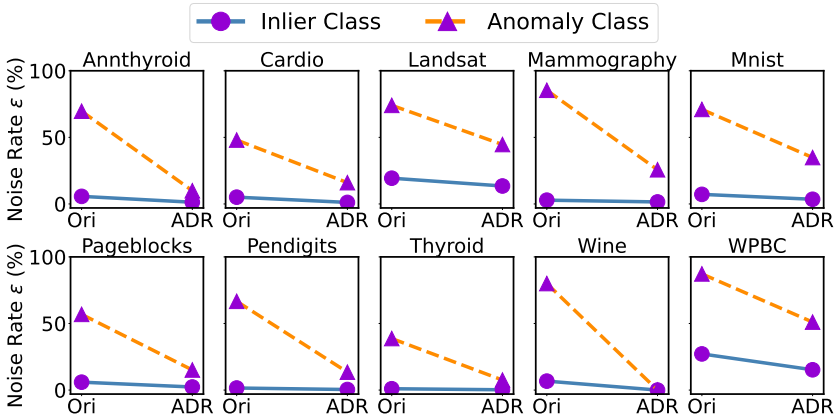


Fig. 6. Change in noise rate per class from original labels (Ori) to selected labels after agreement, disagreement, and refurbishment (ADR) in Epoch 200.

(2) **UNITY-D** which adopts UNITY's Disagreement sample selection (Sec. 3.2) only, and (3) UNITY-CC which adopts **CONTRASTCORR**, performing only UNITY's label refurbishment. To evaluate **CONTRASTCORR** without UNITY's Agree-to-Disagree selection strategy, we leverage the anomaly

Method	Datasets									
	Annthyroid	Cardio	Landsat	Mammography	Mnist	Pageblocks	Pendigits	Thyroid	Wine	WPBC
IF (Base)	$0.34_{.01}$	$0.54_{.02}$	$0.25_{.00}$	$0.26_{.01}$	$0.33_{.02}$	$0.44_{.01}$	$0.39_{.04}$	$0.61_{.04}$	$0.20_{.00}$	$0.17_{.02}$
UNITY-A	$0.39_{.01}$	$0.57_{.04}$	$\underline{0.38_{.08}}$	$\mathbf{0.35_{.05}}$	$0.32_{.04}$	$0.43_{.01}$	$0.47_{.03}$	$\underline{0.77_{.03}}$	$0.37_{.12}$	$0.24_{.01}$
UNITY-D	$\mathbf{0.78_{.00}}$	$\underline{0.62_{.03}}$	$0.31_{.07}$	$0.15_{.11}$	$0.07_{.03}$	$\underline{0.54_{.02}}$	$0.32_{.01}$	$0.73_{.02}$	$0.20_{.00}$	$0.22_{.04}$
UNITY-CC	$0.48_{.02}$	$0.59_{.03}$	$0.30_{.02}$	$0.21_{.04}$	$\underline{0.41_{.01}}$	$0.53_{.01}$	$\underline{0.49_{.04}}$	$0.73_{.02}$	$\underline{0.53_{.15}}$	$\underline{0.30_{.07}}$
UNITY	$\mathbf{0.78_{.00}}$	$\mathbf{0.65_{.09}}$	$\mathbf{0.40_{.02}}$	$\underline{0.34_{.08}}$	$\mathbf{0.48_{.02}}$	$\mathbf{0.63_{.01}}$	$\mathbf{0.58_{.09}}$	$\mathbf{0.81_{.02}}$	$\mathbf{0.83_{.15}}$	$\mathbf{0.36_{.02}}$

Table 4. Ablation Study of performance (F-1) of UNITY's core components: Agree Sample Selection (UNITY-A), Disagreement Resolver (UNITY-D), and UNITY-CONTRASTCORR (UNITY-CC). The best UNITY component per dataset is **bolded**, second best is underlined. Each component consistently provides utility to UNITY over the initial dirty labels (Base).

scores from the IF detector to rank the initial predictions based on network confidence. We then select 20% of the most confident predicted inliers and 20% of the most confident predicted anomalies to train UNITY's refurbishment strategy. The results are shown in Tab. 4.

While both UNITY-A and UNITY-D tend to contribute a performance gain compared to the initial label generator IF for most datasets, their gains are less than the complete UNITY architecture on 8 out of the 10 datasets. This can be explained by UNITY-A capitalizing on selecting easy but accurate labels, while UNITY-D selects riskier marginal samples. We also observe that UNITY-A and UNITY-D struggle to gain in performance on the initial labels when the datasets contain extreme anomaly label noise such as in WPBC (85%), and Wine (90%). This is expected as both components only select samples with clean labels forcing them to discard most anomaly samples when given a high noise rate.

CONTRASTCORR also improves on the performance of the IF generator for most datasets, however, its largest improvement is evident when the noise rate of the anomaly class becomes larger, as seen with WPBC (85%), and Wine (90%) datasets. This is because CONTRASTCORR refurbishes the noise labels and can thus exploit the signal from the noisy marginal samples that UNITY-A and UNITY-D are forced to discard. However, CONTRASTCORR alone always performs worse than the entire UNITY pipeline. This is due to CONTRASTCORR not receiving marginal samples to learn from, while in the UNITY pipeline, UNITY-D selects clean marginal samples to train CONTRASTCORR allowing for optimized performance.

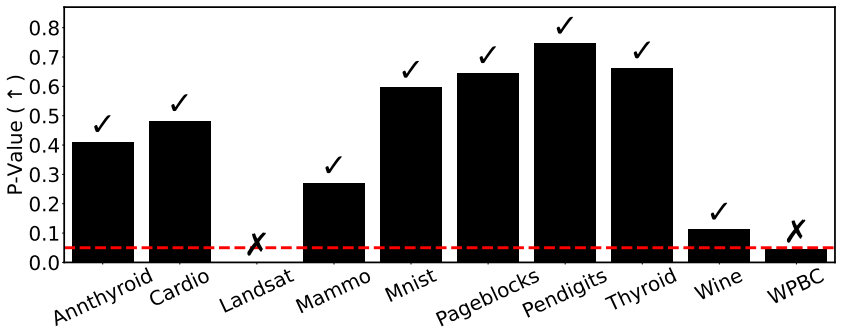


Fig. 7. Paired T-Test (n=10) of F-1 scores for benchmark datasets between UNITY with adaptive vs true threshold. 8/10 datasets point at the adaptive threshold not being significantly different than the true threshold.

6.6 UNITY's Adaptive Threshold Evaluation

To evaluate UNITY's adaptive threshold, we compare the overall F1 performances of UNITY with the adaptive threshold (Sec. 4) vs UNITY with access to the ground truth noise rate to calculate

the threshold. For this comparison, we run both variations for 10 random seeds and perform a paired t-test of the F1 scores, where our null hypothesis is that the two sets of F1 scores will be equal in expectation. We report the p-values for this comparison in Fig. 7. We observe that p-values are larger than 0.05 for 8 out of the 10 datasets ranging from 0.113 to 0.747. We therefore don't have significant evidence to reject the null hypothesis, implying that the adaptive threshold has a similar performance as the threshold with the true noise rate. Meanwhile, for the 2 datasets with p-values less than 0.05, UNITY could gain additional performance when the true noise rate is known. Although UNITY already outperforms all other comparative methods for these datasets without access to the true noise rate.

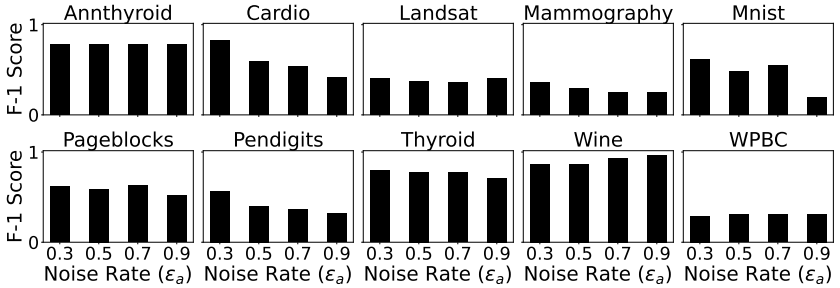


Fig. 8. UNITY for varying noise rates $\bar{\epsilon}_a$ in anomaly class.

6.7 UNITY's Robustness to Label Noise Rates

As demonstrated in Section 6.2, the label noise rate of the anomaly class ϵ_a has a significant impact on the overall anomaly detection performance. We thus evaluate the robustness of UNITY against varying levels of ϵ_a. To focus on the most relevant factor, we vary the noise rate ϵ_a from 30% to 90% in the anomaly class while keeping the noise rate ϵ_i of the inlier class consistent for all datasets. To add realistic noise, when given a desired rate $\bar{\epsilon}_a$, we use the initial rate ϵ_a as the seed. We add label noise to the neighboring noisy anomalies if $\bar{\epsilon}_a > \epsilon_a$, and remove label noise of the marginal noisy anomalies if $\bar{\epsilon}_a < \epsilon_a$. We report F1 scores of UNITY on the new datasets with varying $\bar{\epsilon}_a$ in Fig. 8.

We observe that UNITY remains stable across different anomaly noise rates even when 90% of anomalies are mislabeled for the large majority of the data sets. By combining both clean sample selection and label refurbishment, UNITY selects marginal samples with clean labels and leverages them to further improve the label refurbishment. This allows UNITY to exploit the signal of many samples that previously had noisy labels. The Cardio, Mnist, and Pendigits datasets do however show some instability as the noise rates increase. This can be explained by examining the constant noise rate of the inlier class. The 3 mentioned datasets have a relatively small noise rate for the inlier class as seen in Tab. 2. Therefore UNITY's Agree-to-Disagree sample selection can likely find many inlier samples with clean labels, however, as the noise rate of the anomaly class increases, very few anomalies can be selected as having clean labels. Thus UNITY's CONTRASTCORR may overfit the inlier class, resulting in weaker refurbishment performance.

6.8 UNITY's Runtime Efficiency

We measure the runtime of UNITY, confirming that it matches our time complexity analysis in Sections 3, 4, and 5. We create a series of data subsets with sample sizes ranging from 10K to 200K of our largest dataset Mammography, and measure the average runtime of UNITY for one epoch. Fig. 9 shows the runtimes for agreement (Sec. 3.1), disagreement (Sec. 3.2), and label refurbishment (Sec. 5) subcomponents, respectively. Consistent with the complexity analysis, the overall runtime of UNITY remains linear as the number of samples increases. As expected, UNITY's Disagree takes

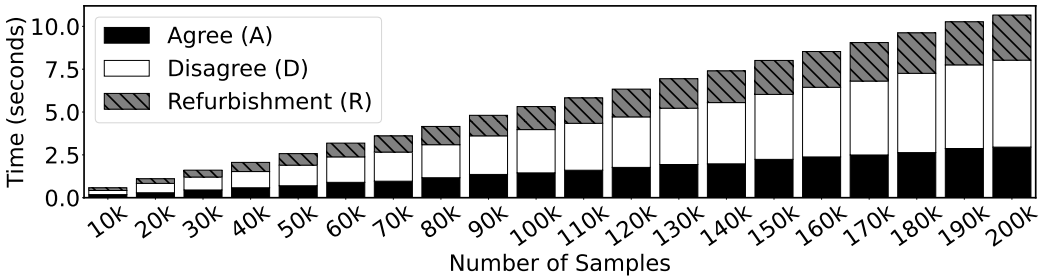


Fig. 9. UNITY runtime efficiency.

about twice as long as UNITY's agreement as the disagreement score is calculated twice, once for each peer network. Overall, UNITY takes only 10 seconds for 200K samples on an A100 GPU, negligible compared to most deep network training. Its lightweight nature makes UNITY suitable for larger datasets and complex peer networks. Since runtime is dominated by model training, and UNITY requires the training of 3 networks, UNITY takes about 3 times longer to run compared to single network approaches.

7 Related Work

Sample Selection. Given a dataset with noisy labels, research on selecting samples with clean labels as a training set predominately focuses on classification tasks [3, 6, 13, 34, 37, 39, 41]. Sample selection is achieved by assigning hard binary weights [13, 37, 39, 41] or soft weights [6, 34] to the samples' loss during network updating. The above methods that target classification assume balanced classes and evenly distributed label noise with a known rate. They leverage signals such as loss or prediction confidence to identify clean samples. A recent method, AutoOD [3], leverages similar intuition of early training loss but applied to anomaly detection.

Label Refurbishment. Label refurbishment methods [7, 33, 40, 47] identify noisy samples using metrics like prediction confidence. Similar to sample selection, these methods typically target classification tasks and work with the assumption of balanced classes and an apriori-known label noise rate. Perhaps closest to our research, SELFIE [36] uses the entropy of network predictions to refurbish samples with consistent predictions. Since anomalies are scarce, they may lack consistent predictions even if they are clean. Instead, UNITY's space transformation via CONTRASTCORR accurately refurbishes the additional anomaly samples, that SELFIE ignores. One variant of SELFIE adds peer networks as per co-teaching [36]. However, when we apply it to the LNL-AD problem, UNITY outperforms SELFIE on all 10 benchmark data sets (Sec. 6). This may be caused by SELFIE aggressively discarding clean marginal samples using peer-network loss; while UNITY's dual disagreement and historical-loss strategy preserves important marginal samples in its selection. Recent work UDAB [40] performs label refurbishment for anomaly detection. They do so assuming anomalies have a greater prediction variance between two networks compared to inliers. Our experiments confirm that UNITY outperforms UADB consistently.

Other Strategies. One recent line of research explores the use of data augmentations for LNL-classification [20, 22, 28]. With anomalies lacking a clear class structure, this type of strategy, when applied to the anomaly problem may risk diluting the true characteristics of the anomalies. Another line of research, namely, the Mixture-of-experts techniques for LNL-AD [45], instead explores the use of many diverse expert sources to help improve performance. They focus on a unique problem setting, different than UNITY, as we instead generate our initial labels from a cheap singular source.

8 Conclusion

We design UNITY, a new learning from noisy labels approach for anomaly detection that, unlike SOTA methods, combines the merits of both sample selection and label refurbishment for dual gain. UNITY's dual deep anomaly classifiers collaboratively select easy samples with clean labels based on prediction agreement and marginal samples with clean labels via disagreement resolution. Thereafter, UNITY transforms the remaining samples into an anomaly-aware embedding space facilitating further separation of marginal clean samples from noisy ones. Our experimental study on 10 benchmark datasets demonstrates that UNITY consistently decreases the noise rate by approximately 2 to 7 folds, and outperforms SOTA LNL methods by 0.31 (0.52 → 0.83) in F-1 Score.

Acknowledgments

This research was supported in part by the National Science Foundation under grants IIS-1815866, IIS-1910880, CSSI-2103832, CNS-1852498, NRT-HDR-2021871 and by the U.S. Dept. of Education under grant P200A180088. Lei Cao is supported by the NSF (DBI-2327954) and Amazon Research Award. We also thank all members of the DAISY group for their input on this research.

References

[1] Firas Abuzaid, Peter Bailis, Jialin Ding, Edward Gan, Samuel Madden, Deepak Narayanan, Kexin Rong, and Sahaana Suri. 2018. MacroBase: Prioritizing Attention in Fast Data. *ACM Trans. Database Syst.* 43, 4, Article 15 (dec 2018), 45 pages. https://doi.org/10.1145/3276463

[2] G. E. P. Box and D. R. Cox. 1964. An Analysis of Transformations. *Journal of the Royal Statistical Society. Series B (Methodological)* 26, 2 (1964), 211–252. http://www.jstor.org/stable/2984418

[3] Lei Cao, Yizhou Yan, Yu Wang, Samuel Madden, and Elke A. Rundensteiner. 2023. AutoOD: Automatic Outlier Detection. *Proc. ACM Manag. Data* 1, 1, Article 20 (may 2023), 27 pages. https://doi.org/10.1145/3588700

[4] Raghavendra Chalapathy and Sanjay Chawla. 2019. Deep learning for anomaly detection: A survey. arXiv preprint arXiv:1901.03407.

[5] Varun Chandola, Arindam Banerjee, and Vipin Kumar. 2009. Anomaly Detection: A Survey. *ACM Comput. Surv.* 41, 3, Article 15 (jul 2009), 58 pages. https://doi.org/10.1145/1541880.1541882

[6] Haw-Shiuan Chang, Erik Learned-Miller, and Andrew McCallum. 2017. Active bias: training more accurate neural networks by emphasizing high variance samples. In *Proceedings of the 31st International Conference on Neural Information Processing Systems* (Long Beach, California, USA) *(NeurIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 1003–1013.

[7] Pengfei Chen, Junjie Ye, Guangyong Chen, Jingwei Zhao, and Pheng-Ann Heng. 2021. Beyond Class-Conditional Assumption: A Primary Attempt to Combat Instance-Dependent Label Noise. *Proceedings of the AAAI Conference on Artificial Intelligence* 35, 13 (May 2021), 11442–11450. https://doi.org/10.1609/aaai.v35i13.17363

[8] Tianyi Chen and Charalampos Tsourakakis. 2022. AntiBenford Subgraphs: Unsupervised Anomaly Detection in Financial Networks. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining* (Washington DC, USA) *(KDD '22)*. Association for Computing Machinery, New York, NY, USA, 2762–2770. https://doi.org/10.1145/3534678.3539100

[9] T. Cover and P. Hart. 1967. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory* 13, 1 (1967), 21–27. https://doi.org/10.1109/TIT.1967.1053964

[10] Yuhao Deng, Chengliang Chai, Lei Cao, Nan Tang, Jiayi Wang, Ju Fan, Ye Yuan, and Guoren Wang. 2024. MisDetect: Iterative Mislabel Detection using Early Loss. *Proc. VLDB Endow.* 17, 6 (May 2024), 1159–1172. https://doi.org/10.14778/3648160.3648161

[11] Andrew Emmott, Shubhomoy Das, Thomas Dietterich, Alan Fern, and Weng-Keen Wong. 2016. A Meta-Analysis of the Anomaly Detection Problem. arXiv:1503.01158 [cs.AI]

[12] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. 2017. On Calibration of Modern Neural Networks. In *Proceedings of the 34th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 70)*, Doina Precup and Yee Whye Teh (Eds.). PMLR, 1321–1330. https://proceedings.mlr.press/v70/guo17a.html

[13] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor W. Tsang, and Masashi Sugiyama. 2018. Co-Teaching: Robust Training of Deep Neural Networks with Extremely Noisy Labels. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems* (Montréal, Canada) *(NeurIPS'18)*. Curran Associates Inc., Red Hook, NY, USA, 8536–8546.

[14] Songqiao Han, Xiyang Hu, Hailiang Huang, Mingqi Jiang, and Yue Zhao. 2022. ADBench: Anomaly Detection Benchmark. Neural Information Processing Systems (NeurIPS).

[15] Peng Jia, Shaofeng Cai, Beng Chin Ooi, Pinghui Wang, and Yiyuan Xiong. 2023. Robust and Transferable Log-based Anomaly Detection. *Proc. ACM Manag. Data* 1, 1, Article 64 (may 2023), 26 pages. https://doi.org/10.1145/3588918

[16] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. 2018. MentorNet: Learning Data-Driven Curriculum for Very Deep Neural Networks on Corrupted Labels. In *ICML*.

[17] Minqi Jiang, Chaochuan Hou, Ao Zheng, Xiyang Hu, Songqiao Han, Hailiang Huang, Xiangnan He, Philip S. Yu, and Yue Zhao. 2023. Weakly Supervised Anomaly Detection: A Survey. arXiv:2302.04549 [cs.LG]

[18] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised Contrastive Learning. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., Red Hook, NY, USA, 18661–18673. https://proceedings.neurips.cc/paper_files/paper/2020/file/d89a66c7c80a29b1bdbab0f2a1a94af8-Paper.pdf

[19] Junnan Li, Richard Socher, and Steven C.H. Hoi. 2020. DivideMix: Learning with Noisy Labels as Semi-supervised Learning. International Conference on Learning Representations.

[20] Junnan Li, Richard Socher, and Steven C. H. Hoi. 2020. DivideMix: Learning with Noisy Labels as Semi-supervised Learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview, OpenReview.net, 14 pages. https://openreview.net/forum?id=HJgExaVtwr

[21] Sainan Li, Qilei Yin, Guoliang Li, Qi Li, Zhuotao Liu, and Jinwei Zhu. 2022. Unsupervised Contextual Anomaly Detection for Database Systems. In *Proceedings of the 2022 International Conference on Management of Data* (Philadelphia, PA, USA) *(SIGMOD '22)*. Association for Computing Machinery, New York, NY, USA, 788–802. https://doi.org/10.1145/3514221.3517861

[22] Yifan Li, Hu Han, Shiguang Shan, and Xilin Chen. 2023. DISC: Learning from Noisy Labels via Dynamic Instance-Specific Selection and Correction. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. OpenReview, OpenReview.net, 24070–24079. https://doi.org/10.1109/CVPR52729.2023.02305

[23] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. 2008. Isolation Forest. 2008 Eighth IEEE International Conference on Data Mining. , 413-422 pages. https://doi.org/10.1109/ICDM.2008.17

[24] Yang Lu, Yiliang Zhang, Bo Han, Yiu-ming Cheung, and Hanzi Wang. 2023. Label-Noise Learning with Intrinsically Long-Tailed Data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 1369–1378.

[25] Donato Malerba, Floriana Esposito, and Giovanni Semeraro. 1996. *A Further Comparison of Simplification Methods for Decision-Tree Induction*. Springer New York, New York, NY, 365–374. https://doi.org/10.1007/978-1-4612-2404-4_35

[26] Olvi L. Mangasarian, W. Nick Street, and William H. Wolberg. 1995. Breast Cancer Diagnosis and Prognosis Via Linear Programming. *Operations Research* 43, 4 (1995), 570–577. https://doi.org/10.1287/opre.43.4.570 arXiv:https://doi.org/10.1287/opre.43.4.570

[27] Anh Nguyen, Jason Yosinski, and Jeff Clune. 2015. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 427–436.

[28] Kento Nishi, Yi Ding, Alex Rich, and Tobias Hollerer. 2021. Augmentation Strategies for Learning With Noisy Labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 8022–8031.

[29] John Paparrizos, Yuhao Kang, Paul Boniol, Ruey S. Tsay, Themis Palpanas, and Michael J. Franklin. 2022. TSB-UAD: an end-to-end benchmark suite for univariate time-series anomaly detection. *Proc. VLDB Endow.* 15, 8 (apr 2022), 1697–1711. https://doi.org/10.14778/3529337.3529354

[30] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems* 32 (2019).

[31] Deep Patel and P S Sastry. 2023. Adaptive Sample Selection for Robust Learning under Label Noise. 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). , 3921-3931 pages. https://doi.org/10.1109/WACV56688.2023.00392

[32] Shebuti Rayana. 2016. ODDS Library. https://odds.cs.stonybrook.edu

[33] Scott E. Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. 2015. Training Deep Neural Networks on Noisy Labels with Bootstrapping. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Workshop Track Proceedings*, Yoshua Bengio and Yann LeCun (Eds.). arXiv, arXiv.org, 11 pages. http://arxiv.org/abs/1412.6596

[34] Mengye Ren, Wenyuan Zeng, Bin Yang, and Raquel Urtasun. 2018. Learning to Reweight Examples for Robust Deep Learning. In *Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 80)*, Jennifer Dy and Andreas Krause (Eds.). PMLR, 4334–4343. https://proceedings.mlr.press/v80/ren18a.html

[35] Lukas Ruff, Robert A. Vandermeulen, Nico Görnitz, Alexander Binder, Emmanuel Müller, Klaus-Robert Müller, and Marius Kloft. 2020. Deep Semi-Supervised Anomaly Detection. In *International Conference on Learning Representations*. https://openreview.net/forum?id=HkgH0TEYwH

[36] Hwanjun Song, Minseok Kim, and Jae-Gil Lee. 2019. SELFIE: Refurbishing Unclean Samples for Robust Deep Learning. Proceedings of the 36th International Conference on Machine Learning. , 5907–5915 pages. https://proceedings.mlr.press/v97/song19b.html

[37] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2021. Robust Learning by Self-Transition for Handling Noisy Labels. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 1490–1500. https://doi.org/10.1145/3447548.3467222

[38] Hwanjun Song, Minseok Kim, Dongmin Park, Yooju Shin, and Jae-Gil Lee. 2023. Learning From Noisy Labels With Deep Neural Networks: A Survey. *IEEE Transactions on Neural Networks and Learning Systems* 34, 11 (2023), 8135–8153. https://doi.org/10.1109/TNNLS.2022.3152527

[39] Hongxin Wei, Lei Feng, Xiangyu Chen, and Bo An. 2020. Combating Noisy Labels by Agreement: A Joint Training Method with Co-Regularization. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). , 13723-13732 pages. https://doi.org/10.1109/CVPR42600.2020.01374

[40] Hangting Ye, Zhining Liu, Xinyi Shen, Wei Cao, Shun Zheng, Xiaofan Gui, Huishuai Zhang, Yi Chang, and Jiang Bian. 2023. UADB: Unsupervised Anomaly Detection Booster. 2023 IEEE 39th International Conference on Data Engineering (ICDE). , 2593-2606 pages. https://doi.org/10.1109/ICDE55515.2023.00199

[41] Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. 2019. How does Disagreement Help Generalization against Label Corruption? International Conference on Machine Learning. , 7164–7173 pages.

[42] Huayi Zhang, Lei Cao, Samuel Madden, and Elke Rundensteiner. 2021. Lancet: labeling complex data at scale. *Proceedings of the VLDB Endowment* 14, 11 (2021).

[43] Huayi Zhang, Lei Cao, Peter VanNostrand, Samuel Madden, and Elke A. Rundensteiner. 2021. ELITE: Robust Deep Anomaly Detection with Meta Gradient. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (Virtual Event, Singapore) *(KDD '21)*. Association for Computing Machinery, New York, NY, USA, 2174–2182. https://doi.org/10.1145/3447548.3467320

[44] Yue Zhao, Zain Nasrullah, and Zheng Li. 2019. PyOD: A Python Toolbox for Scalable Outlier Detection. *Journal of Machine Learning Research* 20, 96 (2019), 1–7. http://jmlr.org/papers/v20/19-011.html

[45] Yue Zhao, Guoqing Zheng, Subhabrata Mukherjee, Robert McCann, and Ahmed Awadallah. 2023. ADMoE: Anomaly Detection with Mixture-of-Experts from Noisy Labels. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence (AAAI'23/IAAI'23/EAAI'23)*. AAAI Press, Red Hook, NY, USA, Article 551, 9 pages. https://doi.org/10.1609/aaai.v37i4.25620

[46] Songzhu Zheng, Pengxiang Wu, Aman Goswami, Mayank Goswami, Dimitris Metaxas, and Chao Chen. 2020. Error-Bounded Correction of Noisy Labels. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, 11447–11457. https://proceedings.mlr.press/v119/zheng20c.html

[47] Songzhu Zheng, Pengxiang Wu, Aman Goswami, Mayank Goswami, Dimitris Metaxas, and Chao Chen. 2020. Error-Bounded Correction of Noisy Labels. In *Proceedings of the 37th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 119)*, Hal Daumé III and Aarti Singh (Eds.). PMLR, proceedings.mlr.press, 11447–11457. https://proceedings.mlr.press/v119/zheng20c.html

[48] Tianyi Zhou, Shengjie Wang, and Jeff Bilmes. 2020. Robust curriculum learning: From clean label detection to noisy label self-correction. International Conference on Learning Representations.

Received July 2024; revised September 2024; accepted November 2024